

A. Appendix

- **SR-1:** [PReserve] **Must Not be Provided If** [No Conflicting Paths Res Others = No]
- **SR-2:** [PCancel] **Must Not be Provided If** [Path State = Occ]
- **SR-3:** [PRelease] **Must Not be Provided If** [Path State = Occ]
- **SR-4:** [BReserve] **Must Not be Provided If** [Track Block Mode = Res]
- **SR-5:** [BCancel] **Must Not be Provided If** [Track Block Mode = Res; Track Block State = Occ]
- **SR-6:** [BRelease] **Must Not be Provided If** [Track Block Mode = Res; Track Block State = Occ]
- **SR-7:** [BRelease] **Must Not be Provided If** [Track Block Mode = Res; Track Block State = Free]
- **SR-8:** [PReserve] **Must Not be Provided If** [No Blocks Are Reserved = No]
- **SR-9:** [PReserve] **Must Not be Provided If** [All Blocks Are Reserved = No]
- **SR-10:** [PCancel] **Must Not be Provided If** [No Blocks Are Occupied = No]
- **SR-11:** [PRelease] **Must Not be Provided If** [No Blocks Are Occupied = No]
- **SR-12:** [Switch] **Must Not be Provided If** [Track Block Mode = Res; Track Block State = Free; Point Position = Ali]
- **SR-13:** [Switch] **Must be Provided If** [Track Block Mode = Res; Track Block State = Free; Point Position = Una]
- **SR-14:** [Switch] **Must be Provided If** [[Track Block Mode = Res; Track Block State = Occ]
- **SR-15:** [Lock] **Must be Provided If** [Point Position = Ali; Point Mode = Unl]
- **SR-16:** [Lock] **Must be Provided If** [Point Position = Una; Point Mode = Unl]
- **SR-17:** [Unlock] **Must Not be Provided If** [Track Block Mode = Res; Track Block State = Free; Point Mode = Lock]
- **SR-18:** [Unlock] **Must Not be Provided If** [Track Block Mode = Res; Track Block State = Occ; Point Position = Ali; Point Mode = Lock]
- **SR-19:** [PReserve] **Must Not be Provided If** [All Points Are Aligned Correctly = No]
- **SR-20:** [PReserve] **Must Not be Provided If** [All Points Are Locked = No]

- **SR-21:** [Green] **Must Not be Provided If** [No Blocks Are Reserved = No]
- **SR-22:** [Green] **Must Not be Provided If** [No Blocks Are Res Others = No]
- **SR-23:** [Green] **Must Not be Provided If** [All Blocks Are Res Oneself = No]
- **SR-24:** [Green] **Must Not be Provided If** [All Points Are Aligned Correctly = No]
- **SR-25:** [Green] **Must Not be Provided If** [All Points Are Locked = No]
- **SR-26:** [Red] **Must Not be Provided** [too late] **If** [No Blocks Are Occupied = No; Signal State = Green]
- **SR-27:** [Red] **Must be Provided If** [No Blocks Are Occupied = No; Signal State = Green]
- **SR-28:** [Red] **Must Not be Provided** [too late] **If** [Route Mode = Can; Signal State = Green]
- **SR-29:** [Red] **Must be Provided If** [Route Mode = Can; Signal State = Green]
- **SR-30:** [Unlock] **Must Not be Provided If** [Signal State = Green]
- **SR-31:** [Switch] **Must Not be Provided If** [Signal State = Green]
- **SR-32:** [BCancel] **Must Not be Provided If** [Signal State = Green]
- **SR-33:** [BRelease] **Must Not be Provided If** [Signal State = Green]

B. Appendix

B.1. Initial Model

B.1.1. Context

```

1  CONTEXT SR0_Path
2  SETS
3  PATH
4  ROUTE
5  CONSTANTS
6  PathConflict
7  Route2InitPath
8  PathSub
9  NullPath
10 AXIOMS
11 axm1: PathConflict ∈ PATH ↔ PATH
12 axm2: ∀p.p ∈ PATH ⇒ (PathConflict ∩ id = ∅)
13 axm3: ∀p.p ∈ PATH ⇒ (PathConflict = PathConflict~)
14 axm4: ∀p.p ∈ PATH ⇒ (finite(PATH))
15 axm5: Route2InitPath ∈ ROUTE → PATH
16 axm6: PathSub ∈ PATH ↔ PATH
17 axm7: ∀p.p ∈ PATH ⇒ (∀p1,p2.p1 ∈ PathSub[{p2}] ∧ p ∉
    PathConflict[{p2}] ⇒ p ∉ PathConflict[{p1}])
18 axm8: ∀p.p ∈ PATH ⇒ NullPath ∈ PathSub[{p}]
19 axm9: NullPath ∈ PATH
20 END
    
```

B.1.2. Machine

```

1  MACHINE SR_M0
2  SEES SR0_Path
3  VARIABLES
4    Route_Req
5    Route_Cel
6    Route2Path
7    Route_Occ
8  INVARIANTS
9    inv1: Route_Req  $\subseteq$  ROUTE
10   inv2: Route_Cel  $\subseteq$  ROUTE
11   inv3: Route2Path  $\in$  ROUTE  $\leftrightarrow$  PATH
12   inv4: Route_Occ  $\subseteq$  ROUTE
13   inv5:  $\forall r1, r2. (r1 \neq r2 \wedge r1 \in \text{dom}(\text{Route2Path}) \wedge r2 \in \text{dom}(\text{Route2Path})) \Rightarrow (\text{PathConflict} \sim [\text{Route2Path}[\{r1\}]] \cap \text{Route2Path}[\{r2\}] = \emptyset)$ 
14   inv6:  $\forall r. r \in \text{Route\_Occ} \Rightarrow (\text{Route2Path}[\{r\}] \neq \emptyset)$ 
15  INITIALISATION:
16  THEN
17    act1: Route_Req :=  $\emptyset$ 
18    act2: Route_Cel :=  $\emptyset$ 
19    act3: Route_Occ :=  $\emptyset$ 
20    act4: Route2Path :=  $\emptyset$ 
21  END
22
23  ATS_Request:
24  ANY r
25  WHERE
26    grd1:  $r \notin \text{Route\_Req}$ 
27  THEN
28    act1: Route_Req := Route_Req  $\cup$  {r}
29  END
30
31  ATS_Cancel:
32  ANY r
33  WHERE
34    grd1:  $r \in \text{Route\_Req}$ 
35  THEN
36    act1: Route_Cel := Route_Cel  $\cup$  {r}
37  END
38
39  Route_Reserve:
40  ANY r
41  WHERE
42    grd1:  $r \in \text{Route\_Req}$ 
43    grd2:  $r \notin \text{Route\_Cel}$ 
44    grd3:  $r \notin \text{dom}(\text{Route2Path})$ 
45  THEN
46    act1: Route2Path := Route2Path  $\cup$  {r  $\rightarrow$  Route2InitPath(r)}
47  END
48
49  Route_Cancel:
50  ANY r
51  WHERE
52    grd1:  $r \in \text{Route\_Req}$ 
53    grd2:  $r \in \text{Route\_Cel}$ 
54    grd3:  $r \in \text{dom}(\text{Route2Path})$ 
55  THEN
56    act1: Route_Req := Route_Req  $\setminus$  {r}
57    act2: Route_Cel := Route_Cel  $\setminus$  {r}
58    act3: Route2Path := {r}  $\triangleleft$  Route2Path
59  END
60
61  Route_Release:
62  ANY r
63  WHERE
64    grd1:  $r \in \text{Route\_Req}$ 
65    grd2:  $r \in \text{dom}(\text{Route2Path})$ 
66    grd3: Route2Path(r) = NullPath
67  THEN
68    act1: Route_Req := Route_Req  $\setminus$  {r}

```

```

69    act2: Route2Path := {r}  $\triangleleft$  Route2Path
70  END
71
72  Train_Enter:
73  ANY r
74  WHERE
75    grd1:  $r \in \text{dom}(\text{Route2Path})$ 
76    grd2:  $r \notin \text{Route\_Occ}$ 
77  THEN
78    act1: Route_Occ := Route_Occ  $\cup$  {r}
79  END
80
81  Train_move:
82  ANY r cp sp
83  WHERE
84    grd1:  $r \in \text{Route\_Occ}$ 
85    grd2:  $r \in \text{dom}(\text{Route2Path})$ 
86    grd3:  $cp \in \text{Route2Path}(r)$ 
87    grd4:  $cp \neq \text{NullPath}$ 
88    grd5:  $sp \in \text{PathSub}[\{cp\}]$ 
89  THEN
90    act1: Route2Path(r) := sp
91  END
92
93  Train_Leave:
94  ANY r
95  WHERE
96    grd1:  $r \in \text{Route\_Occ}$ 
97    grd2:  $r \in \text{dom}(\text{Route2Path})$ 
98    grd2: Route2Path(r) = NullPath
99  THEN
100   act1: Route_Occ := Route_Occ  $\setminus$  {r}
101  END

```

B.2. First Iteration

B.2.1. Context

```

1  CONTEXT SR1_Block
2  EXTENDS SR0_Path
3  SETS
4    BLOCK
5  CONSTANTS
6    Path2Block
7    PathReduce
8    PathIncrease
9  AXIOMS
10  axm1: Path2Block  $\in$  PATH  $\leftrightarrow$  BLOCK
11  axm2:  $\forall p. p \in \text{PATH} \Rightarrow (\forall q. q \notin \text{PathConflict}[\{p\}] \Leftrightarrow (\text{Path2Block}[\{p\}] \cap \text{Path2Block}[\{q\}] = \emptyset))$ 
12  axm3: PathReduce  $\in$  (PATH  $\setminus$  {NullPath})  $\rightarrow$  (BLOCK  $\rightarrow$  PATH)
13  axm4:  $\forall p. p \in \text{PATH} \setminus \{\text{NullPath}\} \Rightarrow (\exists b. b \in \text{BLOCK} \Rightarrow \text{PathReduce}(p)(b) \in \text{PathSub}[\{p\}])$ 
14  axm5:  $\forall p. p \in \text{PATH} \setminus \{\text{NullPath}\} \Rightarrow (\exists b. b \in \text{BLOCK} \Rightarrow \text{Path2Block}[\{\text{PathReduce}(p)(b)\}] = \text{Path2Block}[\{p\}] \setminus \{b\})$ 
15  axm6: PPath2Block[\{NullPath\}] =  $\emptyset$ 
16  axm7: PathIncrease  $\in$  PATH  $\rightarrow$  (BLOCK  $\rightarrow$  PATH)
17  axm8:  $\forall p. p \in \text{PATH} \Rightarrow (\exists b. b \in \text{BLOCK} \Rightarrow p \in \text{PathSub}[\{\text{PathIncrease}(p)(b)\}])$ 
18  axm9:  $\forall p. p \in \text{PATH} \Rightarrow (\exists b. b \in \text{BLOCK} \Rightarrow \text{Path2Block}[\{p\}] \cup \{b\} = \text{Path2Block}[\{\text{PathIncrease}(p)(b)\}])$ 
19  END

```

B.2.2. Machine

```

1  MACHINE SR_M1
2  REFINES SR_M0
3  SEES SR1_Block
4  VARIABLES

```

```

5  ...
6  Block2Route
7  Route2OccPath
8  INVARIANTS
9  inv1: Block2Route ∈ BLOCK ↔ ROUTE
10 inv2: Route2OccPath ∈ ROUTE ↔ PATH
11 inv3:  $\forall p \cdot p \in \text{PATH} \Rightarrow (\forall q \cdot q \notin \text{PathConflict}[\{p\}] \Leftrightarrow (\text{Path2Block}[\{p\}] \cap \text{Path2Block}[\{q\}] = \emptyset))$ 
12 INITIALISATION:
13 THEN
14  ...
15  act5: Block2Route :=  $\emptyset$ 
16  act6: Route2OccPath :=  $\emptyset$ 
17 END
18
19 Block_Reserve:
20 ANY r p
21 WHERE
22  grd1: r ∈ Route_Req
23  grd2: r ∉ Route_Cel
24  grd3: r ∉ dom(Route2Path)
25  grd4: p = Route2InitPath(r)
26 THEN
27  act1: Block2Route := Block2Route ∪ {Path2Block(p) ↦ r}
28 END
29
30 Block_Cancel:
31 ANY r
32 WHERE
33  grd1: r ∈ Route_Cel
34  grd2: r ∈ ran(Block2Route)
35 THEN
36  act1: Block2Route := Block2Route ▷ {r}
37 END
38
39 Block_Release:
40 ANY r b
41 WHERE
42  grd1: r ∈ Route_Occ
43  grd2: b ∈ Block2Route~[{r}]
44 THEN
45  act1: Block2Route := {b} ◀ Block2Route
46 END
47
48 Train_Enter:
49 REFINES Train_Enter
50 ANY r
51 WHERE
52  ...
53  grd3: r ∉ dom(Route2OccPath)
54 THEN
55  ...
56  act2: Route2OccPath := Route2OccPath ∪ {r ↦ NullPath}
57 END
58
59 Train_Head_Move:
60 ANY r op b
61 WHERE
62  grd1: r ∈ Route_Occ
63  grd2: r ∈ dom(Route2Path)
64  grd3: b ∈ Path2Block[{Route2Path(r)}]
65  grd4: r ∈ dom(Route2OccPath)
66  grd5: op = Route2OccPath(r)
67  grd6: b ∉ Path2Block[{op}]
68 THEN
69  act1: Route2OccPath(r) := PathIncrease(op)(b)
70 END
71
72 Train_Rear_Move:
73 REFINES Train_Move
74 ANY r p sp op b
75 WHERE

```

```

76  ...
77  grd6: r ∈ dom(Route2OccPath)
78  grd7: op = Route2OccPath(r)
79  grd8: b ∈ Path2Block[{op}] ∩ Path2Block[{p}]
80  grd9: op ≠ NullPath
81  grd10: sp = PathReduce(p)(b)
82  grd11: PathReduce(op)(b) = NullPath ⇒ card(Path2Block[{p}]) ≠ 1
83 THEN
84  ...
85  act2: Route2OccPath(r) := PathReduce(op)(b)
86 END
87
88 Train_Leave:
89 REFINES Train_Leave
90 ANY r
91  ...
92  grd4: Route2OccPath(r) == NullPath
93 THEN
94  ...
95  act2: Route2OccPath := {r} ◀ Route2OccPath
96 END

```

B.3. Second Iteration

B.3.1. Context

```

1  CONTEXT SR2_Point
2  EXTENDS SR1_Block
3  SETS
4  POS
5  MODE
6  CONSTANTS
7  POINT
8  Point2InitPos
9  Route2Point2Pos
10 Lock
11 Unlock
12 Plus
13 Minus
14 AXIOMS
15 axm1: POINT ⊆ BLOCK
16 axm2: Route2Point2Pos ∈ (ROUTE ↔ POINT) → POS
17 axm3: Point2InitPos ∈ POINT → POS
18 axm4: partition(Pos, {Plus}, {Minus})
19 axm5: partition(Mode, {Lock}, {Unlock})
20 END

```

B.3.2. Machine

```

1  MACHINE SR_M2
2  REFINES SR_M1
3  SEES SR2_Point
4  VARIABLES
5  ...
6  Point2Pos
7  Point2Mode
8  INVARIANTS
9  inv1: Point2Pos ∈ POINT → POS
10 inv2: Point2Mode ∈ POINT → MODE
11 inv3:  $\forall r, po \cdot po \in (\text{Block2Route} \sim [\{r\}] \cap \text{POINT}) \wedge r \in \text{ran}(\text{Block2Route}) \Rightarrow (\text{Point2Pos}(po) = \text{Route2Point2Pos}(\{r \mapsto po\}))$ 
12 inv4:  $\forall r, po \cdot po \in (\text{Block2Route} \sim [\{r\}] \cap \text{POINT}) \wedge r \in \text{ran}(\text{Block2Route}) \Rightarrow (\text{Point2Mode}(po) = \text{Lock})$ 
13 INITIALISATION:
14 THEN
15  ...
16  act7: Point2Pos := Point2InitPos

```

```

17  act8: Point2Mode := POINT × {Unlock}
18  END
19
20  Point_Switch:
21  ANY r po
22  WHERE
23    grd1: r ∉ Route_Cel
24    grd2: r ∉ dom(Route2Path)
25    grd3: po ∈ Block2Route~[{r}] ∩ POINT
26    grd4: Point2Pos(po) ≠ Route2Point2Pos({r ↦ po})
27    grd5: Point2Mode(po) = Unlock
28  THEN
29    act1: Point2Pos(po) := Route2Point2Pos({r ↦ po})
30  END
31
32  Point_Lock:
33  ANY r po
34  WHERE
35    grd1: r ∉ Route_Cel
36    grd2: r ∉ dom(Route2Path)
37    grd3: po ∈ Block2Route~[{r}] ∩ POINT
38    grd4: Point2Pos(po) = Route2Point2Pos({r ↦ po})
39    grd5: Point2Mode(po) = Unlock
40  THEN
41    act1: Point2Mode := {po ↦ Lock} ⋈ Point2Mode
42  END
43
44  Point_Unlock_Cancel:
45  ANY r po
46  WHERE
47    grd1: r ∈ Route_Cel
48    grd2: r ∉ Route_Occ
49    grd3: po ∈ Block2Route~[{r}] ∩ POINT
50    grd4: Point2Mode(po) = Lock
51  THEN
52    act1: Point2Mode := {po ↦ Unlock} ⋈ Point2Mode
53  END
54
55  Point_Unlock_Release:
56  ANY r cp sp po
57  WHERE
58    grd1: r ∈ Route_Occ
59    grd2: po ∈ POINT
60    grd3: po ∈ Block2Route~[{r}]
61    grd4: po ∉ Path2Block[{Route2OccPath(r)}]
62    grd5: po ∉ Path2Block[{Route2Path(r)}]
63  THEN
64    act1: Point2Mode(po) := Unlock
65  END
66
67  Block_Release:
68  REFINES Block_Release
69  ...
70  grd5: b ∈ POINT ⇒ Point2Mode(b) = Unlock
71  ...
72  END
73
74  Block_Cancel:
75  REFINES Block_Cancel
76  ...
77  grd4: ∀po. po ∈ Block2Route~[{r}] ∩ POINT ⇒ Point2Mode(
    po) = Unlock
78  ...
79  END
80
81  Route_Reserve:
82  ...
83  grd6: ∀po. (po ∈ (Path2Block[Route2InitPath[{r}]] ∩ POINT))
    ⇒ (Point2Pos(po) = Route2Point2Pos({r ↦ po}))
84  grd7: ∀po. (po ∈ (Path2Block[Route2InitPath[{r}]] ∩ POINT))
    ⇒ (Point2Mode(po) = Lock)
85  ...
    
```

```

86  END
87
    
```

B.4. Third Iteration

B.4.1. Context

```

1  CONTEXT SR3_Signal
2  EXTENDS SR2_Point
3  SETS
4  SIGNAL
5  CONSTANTS
6  Red
7  Green
8  AXIOMS
9  axm1: partition(SIGNAL, {Red}, {Green})
10 END
    
```

B.4.2. Machine

```

1  MACHINE SR_M3
2  REFINES SR_M2
3  SEES SR3_Signal
4  VARIABLES
5  ...
6  Route2Signal ∈ ROUTE → SIGNAL
7  INITIALISATION:
8  THEN
9  ...
10 act9: Route2Signal := ROUTE × {Red}
11 END
12
13 Signal_Green_Reserve:
14 ANY r
15 WHERE
16   grd1: r ∈ dom(Route2Path)
17   grd2: r ∈ Route_Req
18   grd3: r ∉ Route_Cel
19   grd4: Route2Signal(r) = Red
20 THEN
21   act1: Route2Signal := {r ↦ Green} ⋈ Route2Signal
22 END
23
24 Signal_Red_Cancel:
25 ANY r
26 WHERE
27   grd1: r ∈ dom(Route2Path)
28   grd2: Route2Signal(r) = Green
29   grd3: r ∈ Route_Cel
30 THEN
31   act1: Route2Signal := {r ↦ Red} ⋈ Route2Signal
32 END
33
34 Signal_Red_Occupied:
35 ANY r
36 WHERE
37   grd1: r ∈ dom(Route2Path)
38   grd2: Route2Signal(r) = Green
39   grd3: r ∈ dom(Route2OccPath)
40 THEN
41   act1: Route2Signal(r) := Red
42 END
43
44 Signal_Green_Reserve:
45 ANY r
46 WHERE
47   ...
48   grd5: Block2Route~[{r}] = Path2Block[{Route2InitPath(r)}]
49   grd6: r ∉ dom(Route2OccPath)
    
```

```

50  grd7:  $\forall po \cdot (po \in (\text{Path2Block}[\text{Route2InitPath}[\{r\}]] \cap \text{POINT}))$ 
       $\Rightarrow (\text{Point2Pos}(po) = \text{Route2Point2Pos}(\{r \mapsto po\}))$ 
51  grd8:  $\forall po \cdot (po \in (\text{Path2Block}[\text{Route2InitPath}[\{r\}]] \cap \text{POINT}))$ 
       $\Rightarrow (\text{Point2Lock}(po) = \text{Lock})$ 
52  ...
53  END
54
55  Signal_Red_Cancel:
56  ANY r
57  ...
58  grd4:  $r \notin \text{dom}(\text{Route2OccPath})$ 
59  grd5:  $\forall po \cdot (po \in (\text{Path2Block}[\text{Route2InitPath}[\{r\}]] \cap \text{POINT}))$ 
       $\Rightarrow (\text{Point2Mode}(po) = \text{Lock})$ 
60  grd6:  $\forall po \cdot (po \in (\text{Path2Block}[\text{Route2InitPath}[\{r\}]] \cap \text{POINT}))$ 
       $\Rightarrow (\text{Point2Pos}(po) = \text{Route2Point2Pos}(\{r \mapsto po\}))$ 
61  grd7:  $\text{Block2Route}[\{r\}] = \text{Path2Block}[\{\text{Route2InitPath}(r)\}]$ 
62  ...
63  END
64
65  Signal_Red_Occupied:
66  ANY r
67  ...
68  grd5:  $\text{Route2OccPath}(r) = \text{NullPath}$ 
69  ...
70  END
71
72  Point_Unlock_Cancel:
73  REFINES Point_Unlock_Cancel
74  ...
75  grd5:  $\text{Route2Signal}(r) = \text{Red}$ 
76  ...
77  END
78
79  Point_Unlock_Release:
80  REFINES Point_Unlock_Release
81  ...
82  grd6:  $\text{Route2Signal}(r) = \text{Red}$ 
83  ...
84  END
85
86  Point_Switch:
87  REFINES Point_Unlock_Cancel
88  ...
89  grd7:  $\text{Route2Signal}(r) = \text{Red}$ 
90  ...
91  END
92
93  Block_Cancel:
94  REFINES Block_Cancel
95  ...
96  grd5:  $\text{Route2Signal}(r) = \text{Red}$ 
97  ...
98  END
99
100 Block_Release:
101 REFINES Block_Release
102 ...
103 grd6:  $\text{Route2Signal}(r) = \text{Red}$ 
104 ...
105 END

```